MULTI-DISCIPLINARY WING OPTIMIZATION USING A DISCIPLINE-SPECIFIC TRUST-REGION METHOD

A. Orr[1], S. Marques[1], P. Hewitt[1] and T. Robinson[2]

[1]School of Mechanical and Aerospace Engineering
Queen's University Belfast, Belfast, UK, BT9 5AH
s.marques@qub.ac.uk

[2]Rolls-Royce Deutschland Ltd & Co KG, Eschenweg 11,
Dahlewitz, 15827 Blankenfelde-Mahlow, Germany
theresa-dawn.robinson@rolls-royce.com

Keywords: Optimization, Multi-Disciplinary, Trust-Region.

Abstract: Aircraft design requires compromises between a myriad of disciplines in order to obtain an optimized result. Typical TRMMs have treated multi-disciplinary problems as a single, monolithic problem and used a single trust region size. The Discipline Specific Trust Region Method presented in this paper allows the use of different size trust regions for each discipline, enabling more efficient use of the design space by the low-fidelity models. The method is demonstrated for an analytical function and an aero-structural problem, involving CFD aerodynamic, FE analysis for the high-fidelity models and a vortex lattice method and kriging response surface for the low fidelity models.

1 INTRODUCTION

Design optimization is the process of taking an existing initial engineering design and improving its performance using numerical optimization algorithms. The algorithm requires a function which describes the design, called the objective function, and a set of parameters which may be changed to update the design, called design variables. By varying the design variables, the algorithm attempts to find an improved design relative to the initial condition. Constraints can also be applied to the optimization to prevent designs which would not be feasible, and so the best, feasible design can be obtained. The methods used to implement optimization are broadly applicable; the same method can be applied to an aerodynamic problem as can be applied to a structural problem. However, some methods, such as gradient-based methods, cannot operate on discrete variables while others, such as genetic algorithms, can.

Any method which can make optimization more accessible, or accelerate the process is therefore of benefit. Using the basis that most methods are broadly applicable, optimization frameworks can be developed which can take existing solvers for various simulation techniques, and make them work with an optimization algorithm to form an automated method to generate optimal geometry for a given problem.

The method which will be investigated and expanded on throughout this project is the trust-region model-management method of optimization. A more detailed examination of how this method works will be described later in this paper. In brief, the method uses functions of varying fidelity to quickly and effectively find the optimum of a problem [1]. This method uses a lower-fidelity, and therefore less computationally expensive, model to calculate the optimum within a trusted region. The higher fidelity, more computationally

expensive, model is used only occasionally and only to correct and validate the results of the lower fidelity model.

The benefit of this is that as fewer calls are made to the high-fidelity model and most of the optimization is done using the low-fidelity model, then the optimization should converge more quickly. The high-fidelity model is used to correct any inaccuracies in the model, and decide whether any new point selected by the algorithm is closer to optimal than the previous. The method is guaranteed to converge to a local minimum of the high-fidelity model.

In multidisciplinary design optimization problems, there are situations in which each discipline has a low-fidelity model and a high-fidelity model. One method of using trust-region model management with such a problem is to couple the low-fidelity disciplinary models to create a single low-fidelity multidisciplinary model, couple the high-fidelity disciplinary models to create a single high-fidelity multidisciplinary model, and use standard trust-region model-management methods. Such a method would be effective but does not fully exploit the possible properties of the multi-fidelity methods. If, for example, the low-fidelity structural model is highly predictive of the high-fidelity structural model in a certain region of the design space, but the low-fidelity aerodynamic model is not predictive in that same region, the existing method fails to fully exploit the potential of the multi-fidelity method. This paper presents an alternate approach, in which there is a separate trust-region for each discipline, each sized according to the predictive capability of that discipline's low-fidelity model.

2 TRUST REGION

2.1 Conventional Approach

The trust-region model-management method (TRMM) is a numerical optimization method which uses models of varying fidelity to optimize a problem and has been adapted for both multidisciplinary and multi-objective optimization [2]. In general an unconstrained design problem may be formulated as:

$$\text{minimize} f(\mathbf{x}) \tag{1}$$

Where f is the objective function which is to be minimized and $\mathbf{x}$ is a vector of design variables. As mentioned above TRMM is a variable fidelity method. The high-fidelity model is more accurate and usually more computationally expensive than the low-fidelity model which is also less accurate. Most of the optimization is carried out using the low fidelity model $\tilde{f}(\mathbf{x})$. Optimization using the low-fidelity model is only permissible within a prescribed region of the design space where the low-fidelity model can be trusted to represent the high fidelity model, known as a trust region. When this criteria is met, the TRMM is provably convergent to a minimum of the high-fidelity function [1]. As the low-fidelity model is unlikely to be first order accurate throughout the design space the low-fidelity function must be corrected to become first order consistent with the high-fidelity function, a requirement for convergence of the algorithm. As the low-fidelity model is corrected on a per iteration basis the low-fidelity surrogate formed from correcting the low fidelity model can change between subsequent iterations as the algorithm moves through the design space.

For each iteration of the trust-region algorithm, the optimum of the low-fidelity surrogate is found within the trust region and the performance of the low-fidelity surrogate is

assessed. Depending on the outcome of this assessment the step taken in this iteration is accepted or rejected and the trust region is resized, increasing with good performance, decreasing with poor performance. While any function, with corrections, can be used for the low-fidelity function it is desirable to have a low-fidelity model which is representative of the high-fidelity model as this leads to larger trust regions. Larger trust regions result in fewer high-fidelity iterations for convergence to reached.

To solve the general design problem (1) the algorithm solves a series of trust-region subproblems, the design variables used in these functions are the same as those used in the high-fidelity problem this allows convergence without any mapping between two sets of design variables. The κ^{th} iteration of the subproblem takes the form:

$$\begin{aligned} &\text{minimize} \quad \tilde{f}^{\kappa}(\mathbf{x}) \\ &\text{subject to} \quad \|\mathbf{x} - \mathbf{x}_*^{\kappa}\|_\infty \leq \Delta^{\kappa} \end{aligned} \tag{2}$$

Where $\tilde{f}^{\kappa}$ is the subproblem at the κ^{th} iteration, $\mathbf{x}_*^{\kappa}$ is the centre point of the κ^{th} trust region and the solution to the previous subproblem. δ^{κ} denotes the size of the κ^{th} subproblem. The initial size of the trust region for the first iteration, called Δ_0, is chosen by the user. Some previous knowledge of the problem is required to select a good initial size for Δ it should be small big enough to allow some movement but small enough so that the low-fidelity model is not wildly inaccurate. The trust region will eventually size itself correctly throughout the operation of the algorithm. When a solution for the κ^{th} subproblem is found this is called the trial solution and is denoted $\mathbf{x}_t^{\kappa}$, the validity of this trial step and performance of the low-fidelity function is then assessed before the step is accepted or rejected and the region resized. To assess the trial step first the value of the high fidelity function must be calculated at the trial location $\mathbf{x}_t^{\kappa}$, then the trust region ratio for the κ^{th} iteration, ρ^{κ}, is calculated. The trust-region ratio is calculated using:

$$\rho^{\kappa} = \frac{f(\mathbf{x}_*^{\kappa}) - f(\mathbf{x}_t^{\kappa})}{\tilde{f}(\mathbf{x}_*^{\kappa}) - \tilde{f}(\mathbf{x}_t^{\kappa})} \tag{3}$$

The trust-region ratio is the ratio of improvement in the high-fidelity function to the improvement which was predicted by the low -fidelity function. Hence it gives a measure of both the validity of the trial step and also the performance of the low-fidelity function. In general the larger the trust-region ratio the better the performance of low-fidelity function in a minimization problem. The range of values in which ρ^{κ} falls determines the action taken by the algorithm with respect to the trial step and the size of the trust region algorithm:

- $\rho^{\kappa} < 0$ - The value of the high fidelity function has increased at the trial location, the step is rejected and the size of the trust region is decreased, $\Delta^{\kappa+1} = \Delta^{\kappa}/2$
- $\rho^{\kappa} \leq r_1$ - The value of the high-fidelity function has decreased at the trial location however the low-fidelity function is performing poorly, the step is accepted however the trust region size is decreased, $\Delta^{\kappa+1} = \Delta^{\kappa}/2$
- $r_1 < \rho^{\kappa} \leq r_2$ - The value of the high-fidelity function has decreased at the trial location and the low-fidelity function is performing moderately well, the step is accepted and the trust region size remains unchanged, $\Delta^{\kappa+1} = \Delta^{\kappa}$
- $r_2 < \rho^{\kappa}$ - The value of the high-fidelity function has decreased at the trial location and the low-fidelity function is performing well, the step is accepted and the trust region size is increased, $\Delta^{\kappa+1} = \Delta^{\kappa} * 2$.

and:
r_1 and r_2 are the bounds on the value ranges of ρ^κ, where $0 < r_1 < r_2$. The computationally expensive high-fidelity function need only be called once per iteration, and herein lies the advantage of the TRMM.

2.2 Discipline Specific Trust Region Method

While the traditional trust region algorithm has been used for multidisciplinary optimization problems in the past they have been treated as monolithic problems, hence all trust-region dimensions are scaled equally. The resulting trust-region size becomes a trade-off between the actual sizes required by each discipline. Consider a coupled aero-structural wing optimization problem. For this problem the high fidelity structural model will be finite element model and the high fidelity aerodynamics model will use an CFD solver. The low fidelity structural model will be a kriging model (KM) and the low-fidelity aerodynamics model will be the vortex lattice method (VLM). In some areas of the design space VLM will represent CFD very well, however in similar areas KM may give a poor approximation of the FEA solver. The traditional trust-region algorithm will likely shrink the size of the entire trust region so KM can accurately represent the FEA. This prevents the algorithm capitalizing on the fact the VLM is performing well. The multi-disciplinary algorithm sizes each dimension of the trust region independently which allows the trust region to be grown with respect to the model which is performing well while still reducing in size with respect to the model which is performing poorly. Hence the trust region size is maximized, allowing fewer high-fidelity iterations.

The new algorithm uses an aggregated objective function (AOF) which is comprised of multiple objective functions, relating to each discipline, with coupled and uncoupled design variables, the general design optimization problem then becomes of the form:

$$\text{minimize} f_A(\mathbf{x}_c, \mathbf{x}_1, \mathbf{x}_2) = f_1(\mathbf{x}_c, \mathbf{x}_1) + f_2(\mathbf{x}_c, \mathbf{x}_2) \tag{4}$$

Where f_A is the AOF, f_1 is the first objective function and pertains to the first discipline, $\mathbf{x}$ is the vector of design variables exclusive to the first discipline. f_2 is then the second objective function and pertains to the second discipline, with $\mathbf{x}_2$ being the vector of design variables exclusive to the second discipline. $\mathbf{x}_c$ is the vector of design variables which couple both disciplines.

As with the conventional algorithm the corrections are handled additively. The constraints are again handled using an augmented Lagrangian function, the surrogate AOF is formed:

$$\hat{f}(\mathbf{x}_c, \mathbf{x}_1, \mathbf{x}_2) = \hat{f}_1(\mathbf{x}_c, \mathbf{x}_1) + \hat{f}(\mathbf{x}_c, \mathbf{x}_2) \tag{5}$$

Where $\hat{f}_A$,$\hat{f}_1$,$\hat{f}_2$ are the surrogate AOF, the surrogate first objective function and the surrogate second objective function respectively. The optimization subproblem uses the Eq (5) as the objective function. The trust region-ratio is calculated for the entire AOF using Eq. (3), then the trust-region ratio is calculated for each of the component objective functions using Eq. (6).

$$\rho_i^\kappa = \frac{f(\mathbf{x}_{c*}^\kappa \mathbf{x}_{i*}^\kappa) - f(\mathbf{x}_{tc}^\kappa \mathbf{x}_{it}^\kappa)}{\tilde{f}(\mathbf{x}_{c*}^\kappa \mathbf{x}_{i*}^\kappa) - \tilde{f}(\mathbf{x}_{tc}^\kappa \mathbf{x}_{it}^\kappa)} \tag{6}$$

As with the conventional algorithm this is the ratio of improvement of the high-fidelity function to the predicted improvement given by the low-fidelity function. In this case the

i subscript denotes the discipline to which the ratio refers to. Acceptance or rejection of the trial step is decided based on the value of Eq. (3). Δ_i is resized based on the value of the relevant ρ_i^κ where Δ_i is the trust region dimension for each discipline. The trust-region dimension for the coupled variables, Δ_c is taken as the minimum value of Δ_i. Each new step is processed according to:

- $\rho_A^\kappa \leq 0$ - The value of the high fidelity AOF has increased at the trial location, the step is rejected and the size of the trust region is decreased, $\Delta^{\kappa+1} = \Delta^\kappa/2$
- $\rho_A^\kappa > 0$ The value of the high fidelity AOF has decreased at the trial location, the step is accepted and the trust region will be resized.
- $\rho_i^\kappa \leq r_{i1}$ The value of the high-fidelity function for this discipline has decreased at the trial location however the low-fidelity function is performing poorly, the step is accepted however the trust region size is decreased, $\Delta_i^{\kappa+1} = \Delta_i^\kappa/2$
- $r_{i1} < \rho^\kappa \leq r_{i2}$ The value of the high-fidelity function for this discipline has decreased at the trial location and the low-fidelity function is performing moderately well, the step is accepted and the trust region size remains unchanged, $\Delta_i^{\kappa+1} = \Delta_i^\kappa$
- $r_{i2} < \rho^\kappa$ The value of the high-fidelity function for this discipline has decreased at the trial location and the low-fidelity function is performing well, the step is accepted and the trust region size is increased, $\Delta_i^{\kappa+1} = \Delta_i^\kappa * 2$.

and:
$\Delta_c = \text{minimum}\Delta_i$
r_{i1} and r_{i2} being the bounds on the value ranges of ρ_i^κ, where $0 < r_{i1} < r_{i2}$ for its respective discipline.

3 OPTIMIZATION FRAMEWORK

The optimization framework used in this work has been written in MATLAB. It utilizes the discipline-specific trust-region method described above. The CFD grid generation is automated using the *snappyHexMesh* grid generator from OpenFoam [3], the grids are then converted to Fluent. The structural model and analysis are performed by MSc/Nastran. The coupled aero-structural simulation and all relevant data exchanges are handled by MPCCI [4].

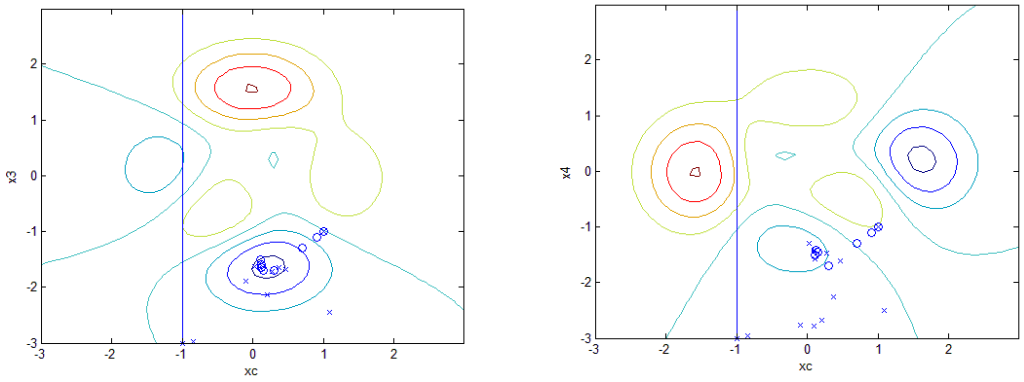
The low-fidelity aero simulation was carried out using Athena Vortex Lattice (AVL), the AVL input geometry is generated using the same design variables as the high fidelity function and passed to AVL for calculation. The low-fidelity structural model is a kriging model generated using the DACE [5] framework from within MATLAB from 50 samples, the sample data was generated using the high-fidelity structural model.

The objective function is minimized using using MATLAB's *fmincon*.

4 RESULTS

4.1 Synthetic Problem I

This section will describe the verification and validation of the algorithm by benchmarking it against a state of the art algorithm for a synthetic numerical test problem. The algorithm against which it will be tested is MATLAB's active set algorithm which utilises a sequential quadratic programming method (SQP).



(a) Peaks function (b) Rotated Peaks Function

Figure 1: Contour plots showing route taken during optimization

The problem is an optimization of MATLAB's peaks function. The first discipline is the peaks function Eq. (7) and the second discipline is a rotation of the peaks function through 90^o Eq. (8). These equations then form the AOF as in Eq. (4).

$$f_3 = 3(1-x_c)^2 \times e^{-x_c^2-(x_1+1)^2} - 10(\frac{x_c}{5} - x_c^3 - x_1^5) \times e^{x_c^2-x_1^2} - \frac{1}{3} \times e^{-(x_c+1)^2-x_1^2} \tag{7}$$

$$f_3 = 3(1-x_2)^2 \times e^{-x_2^2-(-x_c+1)^2} - 10(\frac{x_2}{5} - x_2^3 - x_c^5) \times e^{x_2^2-x_c^2} - \frac{1}{3} \times e^{-(x_2+1)^2-x_2^2} \tag{8}$$

There is a distinct global optimum within each design space, in addition to this each function contains a clearly defined secondary local minimum.

As this is a simple problem it was decided there was little need to construct a low-fidelity approximation of the design space when the additive corrections could be relied upon to generate a first order approximation of the design space within the trust-region on a per iteration basis. The low-fidelity functions are Eq. (9).

$$\tilde{f}_1 = \tilde{f}_2 = 0 \tag{9}$$

This was a constrained optimization problem, the constraint was placed on the coupled variable x_c, the constraint function can be seen in Eqs. (10-12).

$$x_c \geq -1 \tag{10}$$

$$x_1, x_2 \geq -3 \tag{11}$$

$$x_c, x_1, x_2 \leq 3 \tag{12}$$

The vector of design variables $\mathbf{x} = [x_c, x_1, x_2]$ and the value of $\mathbf{x}_0 = [1, 1, 1]$. When the algorithm is initialized the parameters within the algorithm take the values $\Delta_0 = 0.1$, $y_0 = 0$, $\mu_0 = 1 \times 10^-3$,$\tau_0 = 0.25$ and $\eta_0 = 5$. The initial conditions were kept the same for both disciplines.

As mentioned the algorithm is being benchmarked against MATLAB's active set algorithm, it was initialized at the same $\mathbf{x}_0$ and given the same high fidelity functions, Figs. (1(a)) and (1(b)) show the paths each algorithm followed to a local minimum, the 'x' markers indicating the active set algorithm and the 'o' markers indicating the DSTRA. Figure

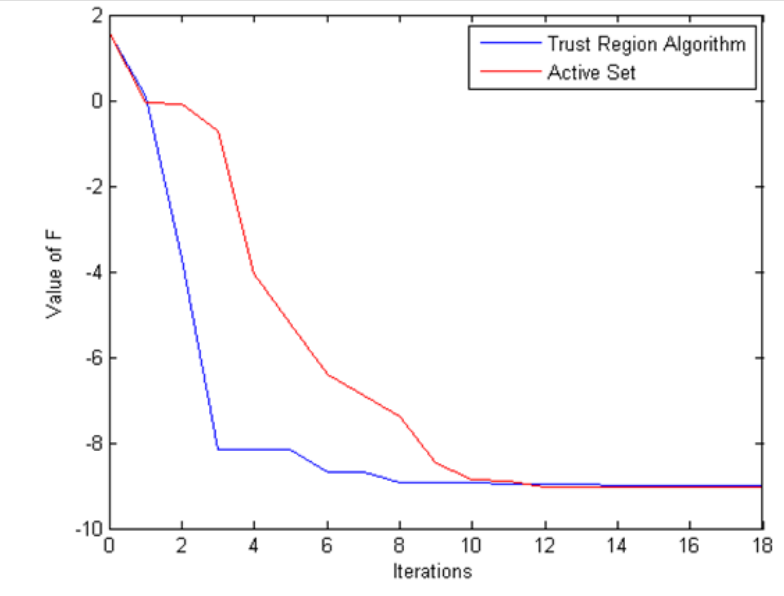


Figure 2: Convergence history of DRSTA and Active set algorithms optimizing the peaks test problem

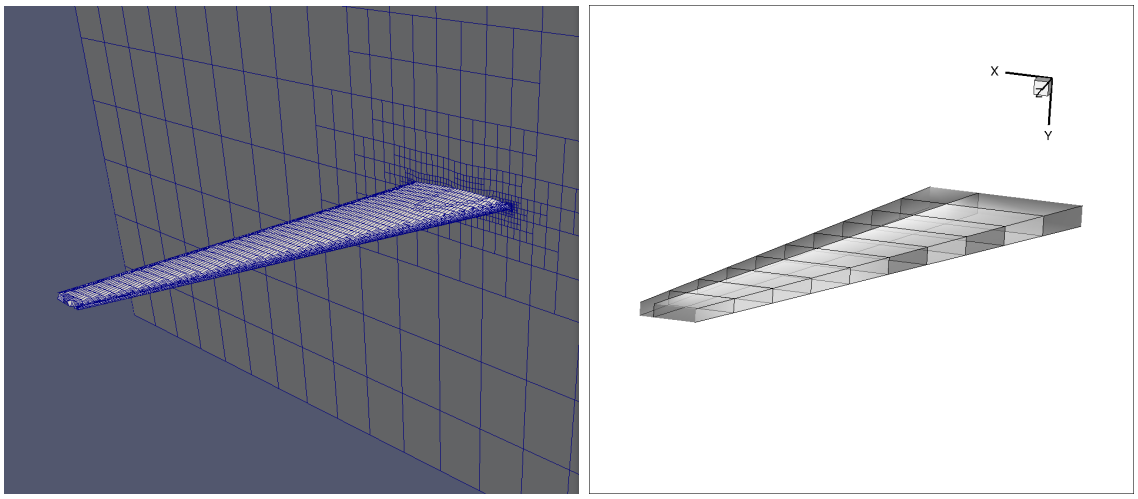
(1(a)) shows both algorithms moving to the same local minimum at $x_c = 0.3$,$x_1 = -1.5$ from the initial position x_0, which is the global minimum for the design space. The initial growth in the trust-region size can be seen with the increase in step size between the first four DSTRA markers, and then the subsequent decrease in size as it closes in on the minimum.

Figure (1(b)) again shows the progress of both algorithms in this case to a local minimum at $x_c = 0.3$,$x_2 = -1.5$, again the initial increase in trust region size can be seen in the first few steps. It is also worth noting the greater rate of descent toward the global minimum in the first discipline with respect to the x_c variable results in the second discipline steering away from the global minimum and finding a local minimum in the second discipline. Again we see both algorithms reach the same local minimum.

Figure (2) shows the convergence history of both algorithms as they progress. It is to be noted that while the DSTRA algorithm converges to the same point as the active set algorithm it does so slightly more slowly, although the both converge in a comparable number of iterations. The initial steps taken by the DSTRA algorithm indicate the algorithm shows significant reduction, although it then takes a further ten iterations for it to fully converge. It should also be noted that the horizontal regions of the DRSTA line on convergence history graph indicate rejected steps, this in turn suggests that a better low-fidelity approximation of the design space would have allowed the algorithm to converge in fewer iterations.

4.2 Aero-Structural Wing Optimization

This section will describe the aero-structural wing optimization problem. The initial wing is as described in Optimization Framework section of this paper. The objective function, eq.(13), was designed to minimize weight and maximize the L/D ratio. Where W is the mass, W_0 is the initial mass, which non-dimensionalises the equation L is Lift and D is



(a) CFD Grid (b) FEA Grid

Figure 3: Initial grids before optimization

drag. Table 2 lists the design variables and the range of values they took.

$$F = \frac{W}{W_0} - \frac{(L/D)}{(L/D)_0} \tag{13}$$

A CFD model is based on the ONERA M6 geometry is used [6]. The wing has a span of 15m, root chord 4.7m and tip chord of 1.7m, in it's initial state the wing has no twist. The typical grid size during the optimization is about 25000 cells. The calculation is carried out at M=0.6 and $\alpha = 0$. Fluent was used to solve the Euler equations, a steady state calculation was used to initialize the calculation for the coupled, transient, calculation, as required by MPCCI. The initial mesh can be seen in fig.(3(a)).

The displacements obtained from the structural analysis, are mapped to the surface of the CFD grid by MPCCI.

The structural model used to compute the displacements for the constraint function can be seen in figure(3(b)). It is a three spar cantilevered wing-box designed scaled for the wing model used in the CFD analysis. It is comprised of 229 elements, table(1) gives a breakdown of the elements used in the construction of the model.

The pressure load on the model is mapped to the surface of the structural model by MPCCI and displacements are computed by MSc/Nastran.

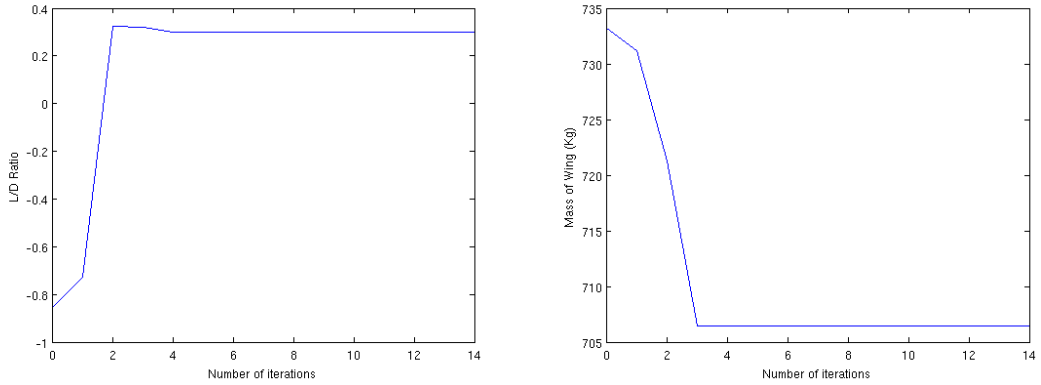
Elements of the FEA model	
CQUAD4	40
CSHEAR	137
CROD	52

Table 1: Elements of the FEA model

Figure (5) shows the convergence history of the algorithm as it progresses through the design space. The initial value of the objective function was 1.8980 and this was driven to 0.7036 by the algorithm. This resulted in an improvement in both the mass of the

Design Variables		
Twist	0^o	$\pm 15^o$
Spar Thickness	3.5×10^{-3}m	$\pm 1.5 \times 10^{-3} m$
Dihedral	0^o	$\pm 3^o$

Table 2: Design Variables used In Wing Optimization



(a) L/D Convergence History (b) Mass Convergence History

Figure 4: Convergence histories of individual objective functions

wing and the L/D ratio. It reached convergence in four iterations. The initial mass of the wing was 733Kg before the optimization, while the final weight of the wing was 706Kg, the convergence history of the mass of the wing can be seen in fig.(4(b)). The initial L/D ratio was -0.8556, the small negative value of the L/D is as a result of the $\alpha = 0$ which produced a small amount of down force and drag. The final value for L/D was 0.3005, the converge history can be seen in fig.(4(a)). The changes made by the algorithm can be seen in table(3).

	Initial	Final
Mass	733.22 Kg	706.46 Kg
L/D ratio	-0.8556	0.3005
Twist Angle	0^o	0.2515^o
Spar Thickness	3.5×10^{-3}m	2.0×10^{-3}m
Dihedral	0^o	1.25^o

Table 3: Comparison of values before and after optimization

5 CONCLUSIONS & OUTLOOK

It has been demonstrated in this paper that the discipline specific trust region method is capable of optimizing synthetic test problems comparably to state of the art algorithms in MATLAB's active set algorithm. Finding the same minimum in slightly fewer iterations than MATLAB's algorithm.

It has also been demonstrated that the algorithm can be used to optimize physical problems, specifically that of optimizing the ONERA M6 wing. The algorithm gave a reduc-

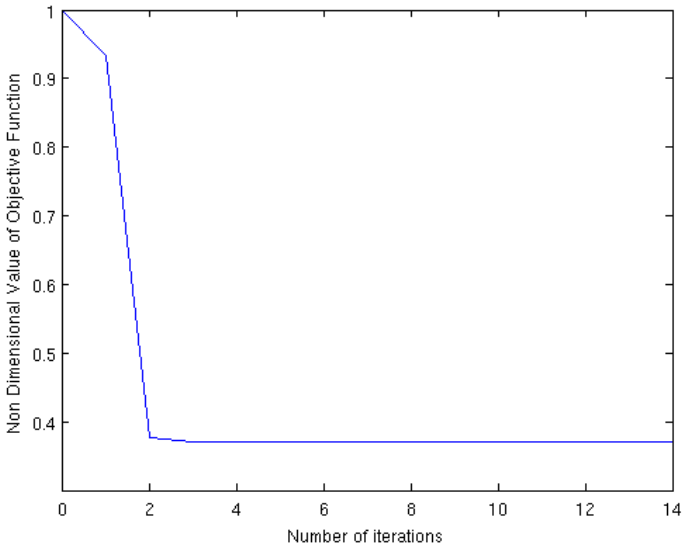


Figure 5: Convergence history of DSTRA for wing optimization problem

tion in weight and an increase in L/D ratio compared to the original wing. In future the algorithm will be used on further physical problems with increased complexity.

6 ACKNOWLEDGEMENTS

This work was funded by the Department for Education and Learning.

7 REFERENCES

[1] Andrew R. Conn, N. I. M. G. and Toint, P. L. (2000). *Trust Region Methods*. SIAM.

[2] Orr A, R. T. (2012). Discipline-specific trust region sizes for variable-fidelity multi-disciplinary optimization. In *AIAA 2012-1930*.

[3] *OpenFOAM v2.1.0 Documentation*.

[4] *MPCCI v4.2.1 Documentation*.

[5] Nielsen, H. B. (2005). Surrogate models: Kriging, radial basis functions, etc. Technical University of Denmark.

[6] Schmitt, V. and Charpin, F. (1979). Pressure distributions on the onera-m6-wing at transonic mach numbers. Experimental Data Base for Computer Program Assessment Report of the Fluid Dynamics Panel Working Group 04, AGARD. AGARD AR 138,.